Theoretical Computer Science 411 (2010) 1685–1697

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

Average-case analysis for the MAX-2SAT problem☆

Osamu Watanabe [a], Masaki Yamamoto [b,*]

[a] Department of Math. and Comp. Sci., Tokyo Inst. of Technology, Japan

[b] Department of Mathematical Sciences, School of Science, Tokai University, Japan

ARTICLE INFO

Article history:
Received 28 February 2007
Received in revised form 19 October 2009
Accepted 25 December 2009
Communicated by A. Razborov

Keywords:
Average case analysis
Planted solution model
Belief propagation
MAX-2SAT

ABSTRACT

We propose a simple probability model for MAX-2SAT instances for discussing the average-case complexity of the MAX-2SAT problem. Our model is a "planted solution model", where each instance is generated randomly from a target solution. We show that for a large range of parameters, the planted solution (more precisely, one of the planted solution pairs) is the optimal solution for the generated instance with high probability. We then give a simple linear-time algorithm based on a message passing method, and we prove that it solves the MAX-2SAT problem with high probability for random MAX-2SAT instances under this planted solution model for probability parameters within a certain range.

1. Introduction

We discuss the average-case analysis of the difficulty of MAX-SAT problems. In particular, we consider the MAX-2SAT problem, the simplest variation of MAX-SAT problems, where input CNF formulas are restricted to those consisting of only clauses with *two* literals. MAX-SAT problems are well known as typical NP-type hard optimization problems, and it is known that even the MAX-2SAT problem is NP-hard, though the 2SAT problem is in P. Furthermore, it is also proved [4] that the MAX-2SAT problem is NP-hard to approximate within a certain constant approximation ratio. However, it has been shown that some algorithms/heuristics solve MAX-SAT problems quite well *on average*: [2,3], and by standard SAT solvers [5,14]. Such algorithms may solve MAX-2SAT as well on average. On the other hand, not so much theoretical investigation has been made for the average-case performance of such algorithms, in particular, on MAX-2SAT instances, compared with various detailed studies on SAT instances.

For discussing the average-case complexity of MAX-2SAT problem, we propose one simple probability model for generating MAX-2SAT instances, thereby giving one instance distribution for the MAX-2SAT problem. Our model is one of the planted solution models. That is, we can guarantee (with high probability[1]) that a planted solution is the optimal solution for a generated instance.

We also demonstrate that a simple linear-time algorithm can solve the MAX-2SAT with high probability when input formulas are given under this distribution with probability parameters in a certain range. Our parameter range is for a dense regime; we could prove that our algorithm solves the MAX-2SAT problem with high probability for random formulas with $O(n^{1.5} \ln n)$ clauses. It is an interesting open problem to show some efficient algorithm for sparse formulas. (Somewhat related result has been shown by Scott and Sorkin [13]. They studied random 2CSP instances (including 2SAT) and showed, among other results, some deterministic algorithm solving MAX-2CSP in polynomial time on average for random sparse

☆ A preliminary version of this paper was presented at the 9th Int'l Conf. on Theory and Applications of Satisfiability Testing, 2006.

* Corresponding author.
E-mail addresses: watanabe@is.titech.ac.jp (O. Watanabe), yamamoto@tokai-u.jp (M. Yamamoto).

[1] Throughout this paper, by "with high probability" we mean probability larger than $1 - \delta$ for a given small constant $\delta > 0$. In the precise statements of our results (i.e., Theorem 2.1 and Theorem 3.1), we explicitly state how δ is related to the other parameters.

0304-3975/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2009.12.020

instances, i.e., formulas with linear number of clauses. The authors ask for algorithms solving MAX-2SAT on dense instances. But note that our distribution is different from the one considered in their paper.)

We begin by introducing some notions and notations for discussing SAT and MAX-SAT problems. Throughout this paper, we will use n and m to denote respectively the number of variables and clauses of a given input Boolean formula. We will use $x_1, \ldots, x_n$ for denoting Boolean variables. A *CNF formula* is a conjunction of clauses, a *clause* is a disjunction of literals, and a *literal* is either a Boolean variable or its negation. In particular, a *2CNF formula* is a formula defined as a conjunction of clauses of two literals, where each clause is specified as $(x_i \vee x_j)$, $(x_i \vee \bar{x}_j)$, $(\bar{x}_i \vee x_j)$, or $(\bar{x}_i \vee \bar{x}_j)$, for some $1 \leq i \leq j \leq n$. In this paper, we will assume that clauses are syntactically one of the above four types; e.g., there is no clause like $(x_j \vee \bar{x}_i)$ for some $i < j$. Note that it is possible that a formula has a clause like $(x_i \vee x_i)$, $(x_i \vee \bar{x}_i)$, or $(\bar{x}_i \vee \bar{x}_i)$. (Since $(\bar{x}_i \vee x_i)$ is semantically the same as $(x_i \vee \bar{x}_i)$, we do not allow clauses of this type. Thus, there are altogether $\binom{n}{2} \times 4 + 3n = 2n^2 + n$ clauses.) We will use ℓ_i to denote either x_i or $\bar{x}_i$.

An *assignment* is a function t mapping $\{x_1, \ldots, x_n\}$ to $\{-1, +1\}$; $t(x_i) = +1$ (resp., $t(x_i) = -1$) means to assign true (resp., false) to a Boolean variable x_i. An assignment is also regarded as a sequence $\boldsymbol{a} = (a_1, a_2, \ldots, a_n)$ of ± 1's, where $a_i = t(x_i)$ for each i, $1 \leq i \leq n$. For a given CNF formula F, its *optimal assignment* is an assignment satisfying the largest number of clauses in F. Now our MAX-2SAT problem is to find, for a given 2CNF formula of the above syntax, an optimal assignment for the formula.

We explain our probability model for generating MAX-2SAT instances. This model is defined as a "planted solution model", a method for generating a problem instance so that a target solution, which is also generated in some way, is the answer to this instance (with high probability). In our model, we generate a sequence $\boldsymbol{a} = (a_1, \ldots, a_n)$ uniformly at random; let $\boldsymbol{a}'$ be its *complement assignment* $(-a_1, \ldots, -a_n)$, i.e., an assignment obtained by flipping the sign of all individual assignments. Then we use a pair of $\boldsymbol{a}$ and $\boldsymbol{a}'$ as a *planted solution pair*. For constructing a formula, we generate each clause satisfied by both assignments with probability p, and it is added to the formula. Since there are n^2 such clauses, the number of clauses of this type added to the formula is *on average* pn^2. In order to make the formula unsatisfiable, we also generate each clause that is unsatisfied by $\boldsymbol{a}$ (resp., $\boldsymbol{a}'$) with probability $r < p$. Again *on average* the formula has $rn(n+1)/2$ clauses that are not satisfied by $\boldsymbol{a}$ (resp., by $\boldsymbol{a}'$). Hence, the generated formula has *on average* $pn^2 + rn(n+1)$ clauses and each assignment of the planted solution pair fails to satisfy $rn(n+1)/2$ clauses *on average*. Note that under this generation model, every literal appears with the same probability. As stated below, we prove that if p is large enough, then one of the planted solution pair is indeed the optimal and no other assignment is as well as this optimal assignment. (See Theorem 2.1 for more detailed statement.)

Theorem 1. *For any constant $\delta > 0$ and for sufficiently large n, if $p = \Omega(\ln(n/\delta)/n)$ and $p/r = \Omega(1)$, then for a randomly generated 2CNF formula F from a random planted solution pair, with probability $> 1 - \delta$, one of the planted solution pair is the optimal assignment of F and no other assignment satisfies as many clauses as this assignment.*

Remark 1 (*Alternative Probability Model*). Under the above condition for p, a generated formula has, with high probability, $\Omega(n \log n)$ clauses. This condition is necessary for the above probability model, where each clause is generated independently. On the other hand, for a probability model where the number of occurrences of each literal is fixed, we can relax this condition and discuss random instances with $O(n)$ clauses. Consider, for example, the following distribution. Here we consider all $4n^2$ possible clauses, including, e.g., $(x_i \vee x_j)$ with $i > j$ and $(x_i \vee \bar{x}_i)$. Consider any p and r so that both $pn/2$ and $rn/2$ are integers. Again we assume that a pair of planted solutions is $(1, 1, \ldots, 1)$ and $(-1, -1, \ldots, -1)$. We generate a formula (i.e., a set of clauses) uniformly at random under the constraint that each literal appears exactly $(p + r)n$ times, pn times in clauses consistent with both planted solutions and rn times in clauses consistent with only one of the planted solution pair. More specifically, we use two sequences S_1 and S_2, where S_1 is a sequence of $pn/2$ copies of $x_1, \ldots, x_n$ and S_2 is its random permutation. Then from the first elements in both sequences in order, we make clauses from corresponding pairs of literals in two sequences. For example, if $S_1 = (x_1, x_1, x_2, x_2, x_3, x_3)$ and $S_2 = (x_3, x_2, x_2, x_1, x_3, x_1)$, then we generate clauses $(x_1 \vee x_3)$, $(x_1 \vee x_2)$, ..., $(x_3 \vee x_1)$. Similarly, by using two sequences consisting of $pn/2$ copies of $\bar{x}_1, \ldots, \bar{x}_n$, we generate clauses of the form $(\bar{x}_i \vee \bar{x}_j)$. On the other hand, by using two sequences consisting of $rn/2$ copies of $x_1, \ldots, x_n$ and $\bar{x}_1, \ldots, \bar{x}_n$ respectively, we generate clauses of the form $(x_i \vee \bar{x}_j)$ and $(\bar{x}_i \vee x_j)$. A formula consisting all these clauses contains each literal exactly $(p + r)n$ times. For this probability model, we can show a property similar to the above theorem, with a condition that $p \geq (2 + \varepsilon)r$ and $p \geq c_\varepsilon/n$ for some constant $c_\varepsilon > 0$. Note that the analysis of our algorithm reported in this paper cannot be used for this probability model. (On the other hand, it is unlikely that our algorithm does not work under this probability model.)

Next we introduce a simple message passing algorithm, and we show that its one instance, which runs in time $\mathcal{O}(n+m)$, can solve the MAX-2SAT problem correctly with high probability if input formulas are given by the above planted solution model with parameters p and r within a certain range.

The idea of the algorithm is simple and intuitively clear. Let F be any 2CNF formula. Considers the case that $x_1 = -1$ in the optimal assignment of F. Suppose that there is a clause $(x_i \vee x_1)$ (resp., $(\bar{x}_i \vee x_1)$) in F, which can be restated as $(\bar{x}_i \to x_1)$ (resp., $(x_i \to x_1)$). Thus, in order to satisfy this clause, we must assign -1 to $\bar{x}_i$ (resp., x_i). Such "negative beliefs" are passed to the other literals from x_1 following backwards implication edges, i.e., directed edges corresponding to implications to x_1. Then for each x_i, a "belief for $x_i = +1$" is computed as $b(x_i) - b(\bar{x}_i)$, where $b(\ell_i)$ is the sum of (negative) beliefs that literal ℓ_i received. Next from literals with a negative belief, their negative beliefs are sent to the other literals through implication edges backwards, and then beliefs are updated based on received beliefs. This process is iterated until the signs of all beliefs

get stabled *or* the number of iterations reaches to a given time bound MAXSTEP. Finally, an assignment to individual x_i is computed from the sign of the final belief of x_i. The other case that $x_1 = +1$ (i.e., $\bar{x}_1 = -1$) is equally considered; we can use one of the outputs satisfying more clauses as a solution.

For our theoretical analysis, we consider a simplified version where MAXSTEP $= 2$, which can be implemented to run in $\mathcal{O}(n+m)$ steps. Even for this simple version, we prove that it solves the MAX-2SAT problem with high probability under the planted solution model with nontrivial parameters p and r. (See Theorem 3.1 for more details and precise statement.)

Theorem 2. *For any constant $\delta > 0$ and for sufficiently large n, if $p = \Omega(\sqrt{\ln(n/\delta)/n})$ and $p/r = \Omega(1)$, then for a randomly generated 2CNF formula F from a random planted solution pair, the algorithm yields these two planted solutions (by executing twice with different beliefs for x_1) with probability $> 1 - \delta$.*

Related work and open problems

There are some proposals of algorithms generating MAX-SAT instances for testing MAX-SAT algorithms [8,9,15]. These algorithms first generate or fix a target assignment and generate an instance so that this assignment becomes an optimal solution. Thus, they are regarded as a planted solution model. In fact, our planted solution model is based on the generation algorithm proposed by Yamamoto [15]. Our improvement here is to introduce a planted solution pair and generate clauses based on two symmetric planted solutions. This makes our model much simpler than the one by Yamamoto's algorithm. Furthermore, we can guarantee that the occurrence of all literals are statistically the same, which prevents solvers to use "majority vote" strategy. The same approach has been proposed [1] for generating hard sat. instances for kSAT problems. The important difference here is the point that inconsistent clauses are also added with probability $r < p$. This is for generating unsatisfiable formulas; otherwise, i.e., if only satisfiable formulas were generated, the problem would be trivially easy because the 2SAT problem is in P, which is different from the other kSAT problem $k \geq 3$. One open problem here is to extend our approach to MAX-kSAT problems for $k \geq 3$.

Our message passing algorithm for MAX-2SAT is motivated by a modified belief propagation algorithm for graph partitioning problems [11]. Recently, Pearl's belief propagation [12] has been used for solving several NP-hard problems, e.g., [7]; but it is also reported that the belief propagation may not be appropriate for solving SAT problems, because the role of literals in each clause is not symmetric. Here we ignore positive literals (i.e., literals assigned true) and use messages from only negative literals. On the other hand, while the belief propagation computes messages by some formula based on the underlying probability model, our algorithm computes messages in a straightforward/naive way. It may be possible to improve our algorithm by using more careful method for computing messages.

Our theoretical analysis of the algorithm is for a special case where MAXSTEP $= 2$, i.e., the case where only two updating iterations is allowed. It is easy to see that $p = \Omega(n^{-1/2})$, which is close to the condition of Theorem 2, is necessary; otherwise, a message from x_1 (or $\bar{x}_1$) cannot reach to the majority of literals. On the other hand, computer experiments show that by allowing more iterations, e.g., MAXSTEP $= 20$ (for $n = 5000$), the algorithm works well for much smaller p. An important open question is to develop some method for analyzing the algorithm's execution for large number of iterations.

In general, it would be interesting to see whether there is some efficient algorithm that solves the MAX-2SAT problem on average for much smaller p. It has been known that SAT problems (under the standard planted solution model) are relatively easy if there are enough number of clauses, which may be also true for MAX-SAT problems. Under the parameter range of Theorem 2 (i.e., $p = \Omega(\sqrt{\ln n/n})$, the number of clauses is *on average* $\Theta(n\sqrt{n \ln n})$. On the other hand, our planted solution model can be used as long as $p = \Omega(\ln n/n)$, in which case generated instances have $\Theta(n \ln n)$ clauses on average. This probability distribution may be an interesting target for MAX-2SAT algorithms.

Preliminaries: the Chernoff bound

In this paper, we will use the following version of the Chernoff bound, modified from the one in [6], precisely, from Theorem 2.3 in page 200.

Proposition 1.1. *Let $X_1, \ldots, X_n$ be independent random variables such that $0 \leq X_i \leq c$ for $1 \leq i \leq n$. Let $S = \sum_{1 \leq i \leq n} X_i$ and $\mu = \mathrm{E}[S]$. Then for any $\varepsilon > 0$, we have*

$$\Pr[\, S \leq (1-\varepsilon)\mu \,] \;\leq\; \exp\left(-\frac{\varepsilon^2 \mu}{2c}\right) \quad \cdots (1)$$

On the other hand, for any $\varepsilon > 0$,

$$\Pr[\, S \geq (1+\varepsilon)\mu \,] \;\leq\; \begin{cases} \exp\left(-\frac{\varepsilon^2 \mu}{3c}\right) & \text{if } \varepsilon \leq 1 \quad \cdots (2) \\ \exp\left(-\frac{\varepsilon \mu}{3c}\right) & \text{if } \varepsilon > 1 \quad \cdots (3) \end{cases}$$

Proposition 1.2. *Let $X_1, \ldots, X_n$ be independent random variables such that $0 \leq X_i \leq c$ for $1 \leq i \leq n$. Let $S = \sum_{1 \leq i \leq n} X_i$ and $\mu = \mathrm{E}[S]$. Then for any $\mu' \geq \mu$ and for any $\varepsilon > 0$, we have*

$$\Pr[\, S \geq (1+\varepsilon)\mu' \,] \;\leq\; \begin{cases} \exp\left(-\frac{\varepsilon^2 \mu'}{3c}\right) & \text{if } \varepsilon\mu' \leq \mu \quad \cdots (4) \\ \exp\left(-\frac{\varepsilon \mu'}{3c}\right) & \text{if } \varepsilon\mu' > \mu \quad \cdots (5) \end{cases}$$

Proof. Let $\varepsilon' = \varepsilon(\mu'/\mu) > 0$. Then,

$$\begin{aligned}\Pr[\,S \geq (1+\varepsilon)\mu'\,] &= \Pr[\,S \geq (1+\varepsilon'(\mu/\mu'))\mu'\,] \\ &\leq \Pr[\,S \geq (1+\varepsilon')\mu\,].\end{aligned}$$

From the above proposition,

$$\Pr[\,S \geq (1+\varepsilon')\mu\,] \leq \begin{cases} \exp\left(-\frac{\varepsilon'^2\mu}{3c}\right) \leq \exp\left(-\frac{\varepsilon^2\mu'}{3c}\right) & \text{if } \varepsilon' \leq 1 \\ \exp\left(-\frac{\varepsilon'\mu}{3c}\right) \leq \exp\left(-\frac{\varepsilon\mu'}{3c}\right) & \text{if } \varepsilon' > 1. \end{cases}$$

Combining these two inequalities, this proposition follows. □

2. A planted solution model for MAX-2SAT

In this section, we define a planted solution model for MAX-2SAT, our probability distribution on instances of MAX-2SAT. More specifically, for a given $n \geq 1$, we describe a way of generating a 2CNF formula over n variables $x_1, \ldots, x_n$.

Consider any assignment $(a_1, \ldots, a_n)$, where $a_i \in \{-1, +1\}$ for $1 \leq i \leq n$, to variables $(x_1, \ldots, x_n)$, and its complement assignment $(-a_1, \ldots, -a_n)$, i.e., an assignment obtained by flipping all values of $(a_1, \ldots, a_n)$. Such a pair of assignments is used as a planted solution pair. For any planted solution pair, a clause is called *consistent* with the planted solution if it is satisfied by both of two assignments of the planted solution, and it is called *partially inconsistent* with the planted solution if it is not satisfied by one of them. (Note that any clause is satisfied by at least one of the planted solution pair.) Now we generate a 2CNF formula as follows: First generate a planted solution pair uniformly at random. Then each clause of the form $(\ell_i \vee \ell_j)$, where $i \leq j$, is added to the formula, with probability p if it is consistent with the planted solution and with probability r if it is partially inconsistent with the planted solution; see below for the case $i = j$.

Remark 2. Recall that ℓ_i denotes a literal either x_i or $\bar{x}_i$. As explained in Introduction, we consider only clauses of the form $(\ell_i \vee \ell_j)$ for some $1 \leq i \leq j \leq n$. For simplifying our analysis, we define our planted solution model so that clauses $(x_i \vee x_i)$ and $(\bar{x}_i \vee \bar{x}_i)$ are respectively added to a formula with probability r, whereas a clause $(x_i \vee \bar{x}_i)$ is generated with probability p. Note that clauses like $(x_i \vee \bar{x}_i)$ are satisfied by any solution. Thus, they can be ignored when discussing the optimality of solutions.

Here in order to simplify our discussion, we will explain with one fixed planted solution, a pair of all $+1$ assignment and all -1 assignment, which we call *our planted solution pair*; we denote them by $\boldsymbol{a}^+$ and $\boldsymbol{a}^-$ respectively. For this solution pair, clauses $(\bar{x}_i \vee x_j)$ and $(x_i \vee \bar{x}_j)$, where $i \leq j$, are consistent and generated with probability p; on the other hand, clauses $(x_i \vee x_j)$ and $(\bar{x}_i \vee \bar{x}_j)$ are partially inconsistent and generated with probability r.

We show below that if $p/r \geq 70$ and $p = \Omega(\ln n/n)$, then for a randomly generated formula F under this planted solution model, one of the planted solution pairs is optimal (and no others) with high probability.

Theorem 2.1. *There exists some constant c_{dist} such that for any $n \geq 1$ and for any $\delta > 0$, if probability parameters p and r satisfy $p/r \geq 70$ and $p \geq c_{\mathrm{dist}} \ln(n/\delta)/n$, then for a randomly generated formula F under our planted solution model with parameters p and r, with probability $\geq 1 - \delta$, one of the two planted solutions is an optimal assignment for F; furthermore, there is no optimal assignment other than the planted solution pair.*

Proof. First, we explain with our planted solution pair, i.e., a pair of all $+1$ assignment $\boldsymbol{a}^+$ and all -1 assignment $\boldsymbol{a}^-$ to n variables $x_1, \ldots, x_n$. Let F be a randomly generated formula for this planted solution pair. Our goal is to show that, with high probability, either $\boldsymbol{a}^+$ or $\boldsymbol{a}^-$ satisfies the most number of clauses in F, which cannot be achieved by any other assignment.

For our discussion, we consider a directed graph $G = (V, E)$ naturally defined as follows: $V = V_+ \cup V_-$, where $V_+ = \{v_{+1}, \ldots, v_{+n}\}$ and $V_- = \{v_{-n}, \ldots, v_{-1}\}$. E consists of two directed edges *corresponding to* a clause $(\ell_i \vee \ell_j)$, where $i \leq j$, in F. For example, for a clause $(x_i \vee x_j)$, E has two directed edges (v_{-i}, v_{+j}) and (v_{-j}, v_{+i}), each of which corresponds to $(\bar{x}_i \to x_j)$ and $(\bar{x}_j \to x_i)$; clauses of the other type define two corresponding directed edges in E similarly. Here note that a clause like $(x_i \vee x_i)$ (resp., $(\bar{x}_i \vee \bar{x}_i)$) has only one corresponding directed edge, namely, the one corresponding to $(\bar{x}_i \to x_i)$ (resp., $(x_i \to \bar{x}_i)$). In order to avoid exceptional cases, we include two directed edges from v_{-i} to v_{+i} for $(x_i \vee x_i)$ (resp., two directed edges from v_{+i} to v_{-i} for $(\bar{x}_i \vee \bar{x}_i)$). Thus, the obtained graph G may have multiple edges. Recall that we do not consider clauses like $(\bar{x}_i \vee x_i)$; also as mentioned in the above remark, we ignore clauses like $(x_i \vee \bar{x}_i)$. Thus, the obtained graph G has no self-loop edge.

Consider any assignment t to $x_1, \ldots, x_n$. We regard this also as an assignment to V; specifically, for each $i \in \{1, \ldots, n\}$, define $t(v_{+i}) = t(x_i)$ and $t(v_{-i}) = -t(v_{+i})$. In general, an assignment t to V satisfying $t(v_{-i}) = -t(v_{+i})$ for all i is called a *legal assignment*; in the following whenever discussing assignments to vertices we will consider only legal assignments. It is easy to see that a clause $(\ell \vee \ell')$ is unsatisfied by t if and only if its two corresponding directed edges are from a vertex assigned $+1$ to a vertex assigned -1, which we call *unsatisfied edges*. That is, the number of unsatisfied clauses is half of that of the unsatisfied edges. Thus, in order to prove the theorem, we estimate the number of unsatisfied edges under an arbitrary legal assignment to V.

Let $G[V_+]$ and $G[V_-]$ denote subgraphs of G induced respectively by V_+ and V_-. First, we estimate the number of unsatisfied edges in $G[V_+]$ and $G[V_-]$; here we will use a well-known fact that a random graph is almost surely an expander.

A directed graph $G' = (V', E')$ is said to be a *d-expander* if for every $S \subset V'$ with $\|S\| \leq \|V'\|/2$, the following holds: $\|E'(S, \overline{S})\| \geq d\|S\|$, and $\|E'(\overline{S}, S)\| \geq d\|S\|$, where $E'(S, \overline{S})$ is the set of edges in E' from vertices in S to vertices in $\overline{S}$. We denote by $\mathcal{G}_{n,q}$ a distribution of graphs $G' = (V', E')$ over n vertices that are generated as follows: for every ordered pair (v'_i, v'_j) of distinct vertices, generate a directed edge (v'_i, v'_j) with probability q as an edge of E'. Recall that when generating a formula F, $G[V_+]$ is generated in this way with probability parameter p, and so is $G[V_-]$. We here show the following expansion property.

Claim 1. *For any n, any $\delta' > 0$, and any $\varepsilon' > 0$, if $q \geq (2/\varepsilon'^2)\ln(4en/\delta')/n$, then for a random directed graph $G' \in \mathcal{G}_{n,q}$, with probability $\geq 1 - \delta'$, $G' = (V', E')$ is a $(1/2 - \varepsilon')qn$-expander.*

Proof of the Claim. For any n, $\delta' > 0$, and $\varepsilon' > 0$, consider any q satisfying the condition of the claim. Let S be a subset of V' with size at most $n/2$. Let $\mathrm{Bad}(S)$ be an event that S does not meet the condition of a d-expander with $d = (1/2 - \varepsilon')qn$, i.e., either $\|E'(S, \overline{S})\| < d\|S\|$ or $\|E'(\overline{S}, S)\| < d\|S\|$ holds. We estimate the upper bound of $\Pr[\, \|E'(S, \overline{S})\| < d\|S\| \,]$. Note that the value of $\Pr[\mathrm{Bad}(S)]$ is at most two times of this value. This is done by Proposition 1.1(1) in the following way: Fix $S \subset V'$ such that $\|S\| \leq n/2$. For all pairs of $u \in S$ and $v \in \overline{S}$, we introduce independent random variables $Y_{u,v}$ such that $\Pr[Y_{u,v} = 1] = q$ and $\Pr[Y_{u,v} = 0] = 1 - q$. Let $Y = \sum_{u \in S, v \in \overline{S}} Y_{u,v}$; that is, $Y = \|E'(S, \overline{S})\|$. Note that $\mathrm{E}[Y] = q(n - \|S\|)\|S\|$; hence, we have

$$\begin{aligned}\mathrm{E}[Y] - d\|S\| &= q(n - \|S\|)\|S\| - d\|S\| \\ &\geq (q(n/2) - d) \cdot \|S\| = \varepsilon' qn\|S\| > 0.\end{aligned}$$

Then from Proposition 1.1(1), it follows

$$\begin{aligned}\Pr[\,\mathrm{Bad}(S)\,] &\leq 2\Pr\left[\, \left\|E'\left(S, \overline{S}\right)\right\| < d\|S\| \,\right] \\ &= 2\Pr[\,\mathrm{E}[Y] - Y > \mathrm{E}[Y] - d\|S\|\,] \\ &< 2\exp\left(-\frac{(\mathrm{E}[Y] - d\|S\|)^2}{2\mathrm{E}[Y]}\right) \\ &\leq 2\exp\left(-\frac{(q(n/2) - d)\|S\|)^2}{2q(n - \|S\|)\|S\|}\right) \\ &\leq 2\exp\left(-\frac{\varepsilon'^2 qn}{2} \cdot \|S\|\right).\end{aligned}$$

Since the probability that G' is not d-expander is the probability that $\mathrm{Bad}(S)$ holds for some $S \subset V'$, $\|S\| \leq n/2$, we have

$$\begin{aligned}\Pr\left[\bigcup_S \mathrm{Bad}(S)\right] &\leq \sum_S \Pr[\,\mathrm{Bad}(S)\,] \\ &\leq \sum_{s=1}^{n/2} \binom{n}{s} 2\exp\left(-\frac{\varepsilon'^2 qn}{2} \cdot s\right) \\ &\leq \sum_{s=1}^{n/2} 2\left(\frac{en}{s} \cdot \exp\left(-\frac{\varepsilon'^2 qn}{2}\right)\right)^s \\ &\leq \sum_{s=1}^{n/2} 2\left(en \cdot \exp\left(-\frac{\varepsilon'^2 qn}{2}\right)\right)^s.\end{aligned}$$

Then the claim holds because the last one is bounded by δ', which is argued as follows: From the assumption that $q \geq (2/\varepsilon'^2)\ln(4en/\delta')/n$, we have $en \cdot \exp(-\varepsilon'^2 qn/2) \leq \delta'/4$; hence, we have $\sum_s (en \cdot \exp(-\varepsilon'^2 qn/2))^s \leq \delta'/2$. □(Proof of the Claim)

Since $G[V_+]$ (resp., $G[V_-]$) can be regarded as a random graph from $\mathcal{G}_{n,p}$, and $p \geq c\ln(n/\delta)/n$ for some sufficiently large constant c by our assumption, we may assume from the above claim that $G[V_+]$ and $G[V_-]$ are $(1/2 - \varepsilon')pn$-expanders for some ε', $0 < \varepsilon' < 1/2$, which will be fixed at the end. That is, for each $U \in \{V_+, V_-\}$ and for every $S \subset U$, we have

$$\begin{aligned}&\|E(S, U - S)\| \geq (1/2 - \varepsilon')pn\|S\|, \quad \text{and} \\ &\|E(U - S, S)\| \geq (1/2 - \varepsilon')pn\|S\|.\end{aligned}$$

Now consider any legal assignment t to V that is different from our two planted solutions. By h we denote the number of unsatisfied edges of G under t. On the other hand, let $h_0 = \min\{|E \cap (V_+ \times V_-)|, |E \cap (V_- \times V_+)|\}$; that is, h_0 is the number

of unsatisfied edges by a better assignment among our two planted solutions. From now on, we estimate h and show that $h > h_0$ with high probability.

Let A_+ and B_+ be subsets of V_+ assigned $+1$ and -1 respectively under t; on the other hand, let A_- and B_- be subsets of V_- assigned -1 and $+1$ respectively. Note that $|A_+| = |A_-|$ and $|B_+| = |B_-|$, and let a and b be the number of $|A_+|$ and $|B_+|$ respectively; we may assume that $a, b \geq 1$. In the case of $a \leq b$, we show that

$$h \ > \ \|E \cap (V_- \times V_+)\| \tag{1}$$

holds with high probability. In the other case, i.e., $a \geq b$, we show that $h > \|E \cap (V_+ \times V_-)\|$ holds with high probability. Then we can conclude $h > h_0$. Below we will consider only the former case, i.e., the case $a \leq b$.

For edges in V_+ and V_-, we see from the above expansion property that the number of unsatisfied edges in each of V_+ and V_- is respectively at least $(1/2-\varepsilon')pn \cdot a$; that is, $\min\{\|E(A_+, V_+ - A_+)\|, \|E(B_-, V_+ - B_-)\|\} \geq (1/2-\varepsilon')pna$. Consider then edges between V_+ and V_-; here we estimate only the number of unsatisfied edges from V_- to V_+. Since any unsatisfied edge is from a $+1$ vertex to a -1 vertex, unsatisfied edges are those from B_- to B_+. Thus, we have $h \geq (1-2\varepsilon')pna + |E \cap (B_- \times B_+)|$, where we decompose the last term as follows.

$$\begin{aligned}
&\|E \cap (B_- \times B_+)\| \\
&= \|E \cap (V_- \times V_+)\| - \|E \cap (A_- \times V_+)\| - \|E \cap (B_- \times A_+)\| \\
&\geq \|E \cap (V_- \times V_+)\| - \|E \cap (A_- \times V_+)\| - \|E \cap (V_- \times A_+)\|.
\end{aligned}$$

Hence, for our goal (1), it suffices to show that $(1-2\varepsilon')pna - \|E \cap (A_- \times V_+)\| - \|E \cap (V_- \times A_+)\|$ is positive. We need the following claim similar to the previous one:

Claim 2. *Let $G = (V, E)$ be a random graph constructed from F generated by our planted solution model with parameters p and r. For any $\delta'' > 0$ and for any $\varepsilon'' > 1/10$, let $0 \leq r' \leq 1$ be a real number such that $r' \geq (6/\varepsilon'')\ln(2en/\delta'')/n$. If $r \leq r'/10$, then the probability that $\|E \cap (A_- \times V_+)\| \geq (1+\varepsilon'')r'(n+1)a$ occurs for some assignment such that $a \leq b$ is bounded by δ''. The same statement also holds for $\|E \cap (V_- \times A_+)\|$.*

Proof of the Claim. The argument is almost the same as Claim 1; here we explain some important points. Consider any assignment t, and let it be fixed for a while. We let w.l.o.g. $A_- = \{v_{-1}, \ldots, v_{-a}\} \subset V_-$; let $I_1 = \{1, \ldots, a\}$ and $I_2 = \{a+1, \ldots, n\}$. For any $i \in I_1$, we introduce the following random variables. For any $j \in \{1, \ldots, n\} \setminus \{i\}$, let X_{ij} be a random variable such that $X_{ij} = 1$ if $(v_{-i}, v_{+j}) \in E$, and $X_{ij} = 0$ otherwise. We let X_{ii} be a random variable such that $X_{ii} = 2$ if two edges (v_{-i}, v_{+i}) corresponding to $(x_i \vee x_i)$ are in E, and $X_{ii} = 0$ otherwise. Also define $Y_{ij} = X_{ij} + X_{ji}$ for $i < j \in I_1$; note that Y_{ij} takes either 0 or 2 because $X_{ij} = X_{ji}$. Then define $X = \sum_{i \in I_1, j \in I_2} X_{ij} + \sum_{i \in I_1} X_{ii} + \sum_{i<j \in I_1} Y_{ij}$ so that $X = \|E \cap (A_- \times V_+)\|$. Note that all random variables appearing in the definition of X are mutually independent. Thus by (4) and (5) of Proposition 1.2, we have, for any $\varepsilon'' > 0$, that

$$\begin{aligned}
\Pr[\, \|E \cap (A_- \times V_+)\| \geq (1+\varepsilon'')\mu' \,] \ &= \Pr[\, X \geq (1+\varepsilon'')\mu' \,] \\
&\leq \begin{cases} \exp\left(-\frac{\varepsilon''^2\mu'}{3\cdot 2}\right) & \text{if } \varepsilon'' \leq \mu/\mu', \\ \exp\left(-\frac{\varepsilon''\mu'}{3\cdot 2}\right) & \text{if } \varepsilon'' > \mu/\mu', \end{cases}
\end{aligned}$$

where $\mu = ra(n-a) + 2ra + 2ra(a-1)/2 = ra(n+1)$ and $\mu' = r'a(n+1)$. Since $\mu/\mu' = r/r' \leq 1/10$, for any $\varepsilon'' > 1/10$,

$$\Pr[\, \|E \cap (A_- \times V_+)\| \geq (1+\varepsilon'')\mu' \,] \ \leq \exp\left(-\frac{\varepsilon''\mu'}{3\cdot 2}\right).$$

Thus, we have

$$\Pr[\, \|E \cap (A_- \times V_+)\| \geq (1+\varepsilon'')r'(n+1)a \,] \ \leq \exp\left(-\frac{\varepsilon''r'(n+1)a}{6}\right) \ \leq \ \exp\left(-\frac{\varepsilon''r'na}{6}\right).$$

Now by considering all possible assignments, or more simply all possible A_-, we can bound the probability that $\|E \cap (A_- \times V_+)\| \geq (1+\varepsilon'')r'(n+1)a$ occurs for some assignment. The derivation of the bound is the same as Claim 1 and omitted here. □(Proof of the Claim)

From this claim, (with high probability) we have for any $0 < a \leq n/2$,

$$(1-2\varepsilon')pna - \|E \cap (A_- \times V_+)\| - \|E \cap (V_- \times A_+)\| \ > \ ((1-2\varepsilon')pn - 2(1+\varepsilon'')r'(n+1))a.$$

Here by letting $\varepsilon' = \varepsilon'' = 1/8$, we can show that the right-hand side is positive for $n \geq 1$ if $p/r' \geq 7$. Thus, for any δ', δ'' with $\delta' = \delta''$, and for any p, r with $p \geq 2 \cdot 8^2 \cdot 7 \cdot \ln(4en/\delta')/n$ and $p/r \geq 70$, if we set $r' = p/7$, then we have $p \geq (2/\varepsilon'^2)\ln(4en/\delta')/n$, $r' = p/7 \geq 2 \cdot 8^2 \cdot \ln(4en/\delta')/n \geq (6/\varepsilon'')\ln(2en/\delta'')/n$, and $r'/r \geq 10$. That is, we obtain all the conditions necessary for the two claims. Thus, (with high probability) the left-hand side of the above inequality is positive, concluding that one of the planted solution is optimal.

Finally we check the probability that the above inequality holds for all assignments. We know from Claim 1 that $\|E(A_+, V_+ - A_+)\| + \|E(B_-, V_- - B_-)\| < (1-2\varepsilon')pna$ occurs for some assignment such that $a \leq b$ is at most $2\delta'$. On

procedure MPalgo_for_MAX-2SAT for input $F = C_1 \wedge \cdots \wedge C_m$;
begin
construct $G = (V, E)$,
where $V = \{ v_s : s \in S \}$, and $E = \bigcup_{1 \le k \le m} E(C_k)$;
set $b(v_s)$ to 0 for all $s \in S$;
$b(v_{+1}) \leftarrow +1$; $b(v_{-1}) \leftarrow -1$;
repeat MAXSTEP times **do** {
for each $i \in \{2, \ldots, n\}$ **in parallel do** {

$$\left.\begin{array}{l} b(v_{+i}) \leftarrow \displaystyle\sum_{v_s:(v_{+i},v_s)\in E} \min(0, b(v_s)); \\ b(v_{-i}) \leftarrow \displaystyle\sum_{v_s:(v_{-i},v_s)\in E} \min(0, b(v_s)); \end{array}\right\} - (*1)$$

$$b(v_{+i}) \leftarrow b(v_{+i}) - b(v_{-i}); \quad b(v_{-i}) \leftarrow -b(v_{+i}); \ -(*2)$$

}
if $\mathrm{sign}(b(v_i))$ is stabilized for all $i \in \{2, \ldots, n\}$
then break;
}
output($+1, \mathrm{sign}(b(v_{+2})), \ldots, \mathrm{sign}(b(v_{+n}))$);
end-procedure

Our message passing algorithm: This is an execution for a given 2CNF formula F over variables $x_1, \ldots, x_n$ under the assumption that $x_1 = +1$ (i.e., true); the algorithm should be also executed from the initial value $b(v_{+1}) = -1$. Here we give brief explanation on symbols used in the algorithm (see the text for details): $S = S_+ \cup S_-$, where $S_+ = \{+1, \ldots, +n\}$ and $S_- = \{-n, \ldots, -1\}$. For any $C_k = (\ell_i \vee \ell_j)$, where $i \le j$, $\ell_i = x_i$ or $\bar{x}_i$, and $\ell_j = x_j$ or $\bar{x}_j$, $E(C_k)$ is the set of directed edges corresponding to $(\bar{\ell}_i \to \ell_j)$ and $(\bar{\ell}_j \to \ell_i)$. In case $\ell_i = \ell_j$ and hence $(\bar{\ell}_i \to \ell_j) = (\bar{\ell}_j \to \ell_i)$, $E(C_k)$ is a singleton; otherwise $E(C_k)$ has two edges. Our theoretical analysis is made for the case MAXSTEP $= 2$, that is, the case where the algorithm yields an output after two iterations; for simplifying the analysis, we further assume the following $(*)$: the statements at $(*2)$ are not executed for the first iteration.

Fig. 1. A message passing algorithm for the MAX-2SAT problem.

the other hand, from Claim 2, the probability that $\|E \cap (A_- \times V_+)\| + \|E \cap (V_- \times A_+)\| \ge 2(1 + \varepsilon'')r'(n + 1)a$ occurs for some assignment such that $a \le b$ is at most $2^{\delta''}$. Thus, by choosing $\delta' = \delta'' = \delta/8$, we can show that the probability that the above event fails to hold is at most $\delta/2$. Considering the other case as well, we can conclude that with probability $1 - \delta$, the desired inequality holds for all assignments. □

Remark 3. By looking at the proof of the theorem (i.e., the analysis of its very end), it is easy to see that the condition $p \ge (2 + \varepsilon)r$ suffices for the theorem, while the constant c_{dist} depends on the choice of ε. On the other hand, the coefficient 2 is due to a bit rough estimate for the expansion property at Claim 1 following the standard argument. While we think that it is possible to improve this to any constant larger than 1, we leave it to the interest reader.

Remark 4 (*Proof for the Alternative Probability Model*). As stated in Remark 1, we can prove somewhat stronger statement for more balanced probability models. Consider, for example, the one defined in Remark 1. In this case, since we may assume that the corresponding directed graphs $G[V_+]$ and $G[V_-]$ are both pn-regular, by following a standard argument [10], we can show that the desired expander property holds for these graphs if $p \ge c/n$ for sufficiently large $c > 0$. On the other hand, we know from the assumption that the number of crossing edges (i.e., $\|E \cap (A_- \times V_+)\|$ and $\|E \cap (V_- \times A_+)\|$ respectively) is exactly $2rna$ (for which we do not need any proof like Claim 2). Then by an argument similar to the above, we can show the corresponding statement if $p \ge (2 + \varepsilon)r$ and $p \ge c_\varepsilon/n$, for any $\varepsilon > 0$ and for some constant $c_\varepsilon > 0$.

3. A simple algorithm

For our probability model for the average-case analysis of MAX-2SAT, we show in this section that a simple algorithm can solve MAX-2SAT on average when parameters p and r are in a certain but nontrivial range. The algorithm is a message passing algorithm stated in Fig. 1; this algorithm is motivated by the modified belief propagation algorithm for graph partitioning problems [11].

We explain the outline of the algorithm.[2] First define the meaning of symbols used in the algorithm. The algorithm is executed on a directed graph $G = (V, E)$ that is constructed from a given formula F in essentially the same way as in the proof of Theorem 2.1. with some minor difference. V is a set of $2n$ vertices v_s, $s \in S = \{-n, -(n-1), \ldots, -1, +1, \ldots, +(n-1), +n\}$, and E consists of two directed edges corresponding to each clause $(\ell_i \vee \ell_j)$ of F, where $i < j$; on the other hand,

[2] In this section, we will use i and j to denote unsigned (i.e., positive) indices in $\{1, \ldots, n\}$, whereas s and t will be used for signed indices in S.

only one edge is added to E for each clause of type $(\bar{x}_i \vee x_i)$. Note that graph G has no multiple edge, while it may have some self-loops. (*Cf.* In the proof of Theorem 2.1, a graph has no self-loop while it may have multiple edges.) The algorithm computes a "belief" $b(v_s)$ at each vertex v_s, an integral value indicating whether the Boolean variable $x_{|s|}$ should be assigned true (i.e., $+1$) or false (i.e., -1). More specifically, for an optimal assignment, the algorithm suggests,[3] for each x_i, to assign $x_i = +1$ if the final value of $b(v_{+i})$ is positive and $x_i = -1$ if it is negative. Note that $b(v_{-i}) = -b(v_{+i})$; we may regard $b(v_{-i})$ as a belief for $\bar{x}_i$.

These belief values are initially set to 0 except for one pair of vertices, e.g., v_{+1} and v_{-1} that are assigned $+1$ or -1 initially. In the algorithm of Fig. 1, $b(v_{+1})$ (resp., $b(v_{-1})$) is set to $+1$ (resp., -1), which considers the case that x_1 is true in the optimal assignment. Clearly we need to consider the other case; that is, the algorithm is executed again with the initial assignment $b(v_{+1}) = -1$ and $b(v_{-1}) = +1$, and one of the obtained assignments satisfying more clauses is used as an answer.

Now consider the execution of the algorithm, and explain how beliefs are updated. At each iteration of the execution, the belief of each vertex v_{+i} (resp., v_{-i}) is recomputed based on messages from its neighbor vertices. It should be remarked here that all belief values are updated *in parallel*; that is, updated beliefs are not used when updating the other beliefs in the same iteration, but those computed at the previous iteration are used. Let us see in more detail how beliefs are updated. First look at the computation at $(*1)$ in the algorithm. Here we use only the influence from vertices with negative belief, that is, vertices whose corresponding literals are (currently) assumed to be false. Suppose that at some point, the belief $b(v_s)$ of vertex v_s gets negative; this suggests that the literal $\ell_{|s|}$ corresponding to v_s should be assigned false. Suppose also there is a directed edge (v_{+i}, v_s) from v_{+i} to v_s in E, which means that a clause $(x_i \to \ell_{|s|})$ exists in F. Then for satisfying this particular clause (under the situation that $\ell_{|s|}$ is assigned false) x_i should be assigned also false, which is expressed by adding some negative value to $b(v_{+i})$. This is why at $(*1)$, temporal values $b(v_{+i})$ and $b(v_{-i})$ are computed by summing up all such negative beliefs from their neighbor vertices. These two values are considered as temporal local requirements to the assignment of x_i. Then at $(*2)$ new belief $b(v_{+i})$ is computed as $b(v_{+i}) - b(v_{-i})$. Although this update is based on local relations, we expect intuitively that beliefs get closer to those suggesting the optimal solution by executing this updating process for several times.

This is the outline of our algorithm. Though its detailed implementation is omitted here, it should be noted here that one can implement the algorithm on the standard unit cost RAM model so that each iteration can be done in time $\mathcal{O}(n + m)$. Thus, if the algorithm works for some constant MAXSTEP (as shown in the following analysis), the total running time of the algorithm is $\mathcal{O}(n + m)$.

3.1. Theoretical analysis of the algorithm (for MAXSTEP = 2)

We state our theoretical analysis of the algorithm and prove Theorem 2. Due to some technical reason that will be explained in the proof, we prove only some very restricted case where the algorithm is executed with MAXSTEP $= 2$, i.e., two updating rounds. Obviously the algorithm works better by executing the updating process for many times; but the analysis of such executions is left to our future work. Furthermore, in order to simplify our analysis, we assume the following modification $(*)$: the statements at $(*2)$ are executed only after the second iteration. That is, we what we will prove here is precisely the following theorem.

Theorem 3.1. *There exists some constant c_{algo} such that for any $\delta > 0$, if $p/r \geq c_{\text{algo}}$ and $n \geq c_{\text{algo}} \ln(12n/\delta)/p^2$, then the algorithm (under MAXSTEP $= 2$ and the modification $(*)$) executed with two different initial values for $b(v_{+1})$ (resp., $b(v_{-1})$), yields a pair of all $+1$ and all -1 planted solution with probability $1 - \delta$.*

We will prove this theorem in the following. Roughly speaking, our goal is to show that if $p/r \geq c$ for a sufficiently large constant c, and n is large enough so that $n = \Omega(\ln n/p^2)$ (or equivalently, $p = \Omega(\sqrt{\ln n/n})$ holds, then with high probability the algorithm yields each planted solution from the corresponding initial value of $b(v_{+1})$. Hence, by running the algorithm twice with two different initial values (i.e., $+1$ and -1), we can get two planted solutions. As argued in the previous section, if an input formula F is generated under our planted solution model with parameters $p/r \geq 70$, then one of the two planted solutions is an optimal assignment for F with high probability; thus, by running the algorithm twice with two different initial values, we can obtain the optimal assignment with high probability.

Fix n and fix our planted solution pair to all $+1$ (i.e., true) assignment and all -1 (i.e., false) assignment. We consider the situation where the algorithm is executed with initial values $b(v_{+1}) = +1$ and $b(v_{-1}) = -1$; thus, our goal is to show that the algorithm outputs all $+1$ assignment. We assume that an input formula F is randomly generated following our planted solution model for this planted solution pair with parameters p and r such that $p/r \geq c$ for a sufficiently large constant c. Thus, F is a random variable in our following discussion. Below we introduce some more random variables, all of which depend on the random variable F.

Let G be a graph constructed from F in the algorithm. For any s and t in S, let $E_{s,t}$ be a random variable indicating whether there is an edge from v_s to v_t in G; i.e., $E_{s,t} = 1$ if an edge (v_s, v_t) exists and $E_{s,t} = 0$ otherwise. From the definition of the

[3] In the case that the final value $b(v_{+i}) = 0$ for some $i \in \{2, \ldots, n\}$, the execution is regarded as failure.

graph G, it is clear that random variables $E_{s,t}$ and $E_{s',t'}$ are mutually independent except that it holds $E_{s,t} = E_{-t,-s}$. Also from the choice of our planted solution pair, we may assume that $E_{s,t}$ takes either 0 or 1 as follows:

$$E_{s,t} = \begin{cases} 1, & \text{with prob. } p, \\ 0, & \text{with prob. } 1-p. \end{cases} \quad \text{s.t. } \mathrm{sign}(s) = \mathrm{sign}(t) \tag{2}$$

and

$$E_{s,t} = \begin{cases} 1, & \text{with prob. } r, \\ 0, & \text{with prob. } 1-r. \end{cases} \quad \text{s.t. } \mathrm{sign}(s) \neq \mathrm{sign}(t) \tag{3}$$

For example, $E_{+2,-3} = 1$ if clause $(x_2 \to \bar{x}_3)$ $(= (\bar{x}_2 \vee \bar{x}_3))$ exits in F, which occurs with probability r, and $E_{-1,-4} = 1$ if clause $(\bar{x}_1 \to \bar{x}_4)$ $(= (x_1 \vee \bar{x}_4))$ exits in F, which occurs with probability p.

Consider the status of the algorithm after the first iteration. Note first that vertices v_s whose belief is updated (from 0) is only those having an edge *to* v_{-1} because v_{-1} is only the vertex with negative belief. Let W_+ (resp., W_-) be the set of vertices v_{+j} (resp., v_{-j}) having a directed edge to v_{-1}. Since there is no multiple edge and the statements at $(*2)$ are not executed for the first iteration (due to our technical assumption $(*)$), we have $b(v_s) = -1$ if and only if $v_s \in W_+ \cup W_-$. From our expecting planted solution (i.e., all $+1$ assignment), we may regard those $v_{+j} \in W_+$ as vertices with (currently) wrong belief, i.e., $b(v_{+j}) = -1$. Note also that W_+ and W_- are random variables determined by E (and hence by F). Let Y_+ and Y_- respectively denote the size of W_+ and W_-. In summary, we consider the following random variables for the first iteration. (In order to simplify our analysis, the influence from the vertex v_{+1}, v_{-1} is considered only for the first iteration, and it will be ignored in the second iteration. Thus, here and in the following, we assume that the domain of indices j is $\{2, \ldots, n\}$.)

$$\begin{aligned} W_+ &= \{\, v_{+j} : (v_{+j}, v_{-1}) \in E \,\} = \{\, v_{+j} : b(v_{+j}) = -1 \,\}, \\ W_- &= \{\, v_{-j} : (v_{-j}, v_{-1}) \in E \,\} = \{\, v_{-j} : b(v_{-j}) = -1 \,\}, \\ Y_+ &= \|W_+\|, \text{ and } Y_- = \|W_-\|. \end{aligned}$$

From our random model and choice of parameters (i.e., (2) and (3)), it is easy to see that

$$Y_+ \sim \mathrm{B}(n-1, r) \quad \text{and} \quad Y_- \sim \mathrm{B}(n-1, p). \tag{4}$$

For the second iteration, we consider random variables X_s's, where each X_s is the value of $b(v_s)$ at the second iteration after executing the statements at $(*1)$ and before executing the statements at $(*2)$. For each X_s, we consider further two random variables X_s^+ and X_s^- for summarizing the messages from vertices with positive and negative indices respectively. With $b'(v_t)$ denoting the belief of v_t computed at the first iteration, these random variables are defined as follows. (*Remark.* Recall that we ignore the influence from v_{+1}, v_{-1} in the second iteration, which may cause some difference in the following analysis of each belief; but the difference at each vertex is at most ± 2 and it can be ignored for any sufficiently large n under our choice of parameters p and r.)

$$\begin{aligned} X_s^+ &= \sum_{+j:(v_s, v_{+j}) \in E} \min(0, b'(v_{+j})), \qquad X_s^- = \sum_{-j:(v_s, v_{-j}) \in E} \min(0, b'(v_{-j})), \text{ and} \\ X_s &= \sum_{t:(v_s, v_t) \in E} b'(v_t) = X_s^+ + X_s^-. \end{aligned}$$

Note that each $b'(v_{+j})$ (resp., $b'(v_{-j})$) takes nonzero negative value (in fact, -1) if and only if $+j$ is in W_+ (resp., $-j$ is in W_-). Thus, we have

$$X_s^+ = \sum_{+j \in W_+} -E_{s,+j} \quad \text{and} \quad X_s^- = \sum_{-j \in W_-} -E_{s,-j}. \tag{5}$$

Finally, for each i, define $Z_i = X_{+i} - X_{-i}$; this is the final belief $b(v_{+i})$ computed by the algorithm. Recall that the algorithm determines an output assignment to x_i by the sign of $b(v_{+i})$; hence, our goal is to show that every Z_i is positive with high probability.

Since all random variables Z_i's follow the same distribution, in the following, we will fix some i and consider Z_i; let s to denote either $+i$ or $-i$ only. Also let n' denotes $n-1$.

We first estimate the expectation of Z_i.

Lemma 3.2. $\mathrm{E}[\, Z_i \,] = (p-r)^2 n'$.

Proof. Consider X_{+i}^+, for example. Roughly speaking, from our random model (more specifically, from (2), (4), and (5)), we have $X_{+i}^+ \sim -\mathrm{B}(Y_+, p)$ and $Y_+ \sim \mathrm{B}(n', r)$, which implies $\mathrm{E}[X_{+i}^+] = -prn'$. Here we examine this a bit more carefully.

First note the following derived from (5) and the definition of W_+.

$$X_{+i}^+ = \sum_{+j \in W_+} -E_{+i,+j} = \sum_{k \in \{2,\ldots,n\}} -E_{+i,+k} \cdot E_{+k,-1}.$$

Here a key is the independence between $E_{+i,+k}$ and $E_{+k,-1}$ (due to the condition $k \neq 1$). Thus, the above estimation is derived formally as follows.

$$\mathrm{E}[X^+_{+i}] \;=\; \sum_{k\in\{2,\ldots,n\}} -\mathrm{E}[E_{+i,+k}]\cdot\mathrm{E}[E_{+k,-1}] \;=\; -prn'.$$

By similar independence relations, we also have

$$\mathrm{E}[X^-_{+i}] = -prn', \qquad \mathrm{E}[X^+_{-i}] = -r^2n', \quad \text{and} \quad \mathrm{E}[X^-_{-i}] = -p^2n'.$$

Hence, we have

$$\begin{aligned}\mathrm{E}[Z_i] \;&=\; \Big(\mathrm{E}[X^+_{+i}] + \mathrm{E}[X^-_{+i}]\Big) - \Big(\mathrm{E}[X^+_{-i}] + \mathrm{E}[X^-_{-i}]\Big)\\ &= (p-r)^2n'. \quad \square\end{aligned}$$

It follows from this analysis that if $p - r$ is large enough (i.e., $p - r > (n-1)^{-1/2}$), then Z_i is positive *on average*. Then for proving that Z_i is positive *with high probability* it now suffices to show that their deviation from the average is small with high probability, which is our goal in the following analysis.

The deviation of Z_i from its expectation is caused by the deviation of any of the random variables used for defining Z_i; that is, six random variables Y_+, Y_-, X^+_{+i}, X^+_{-i}, X^-_{+i}, and X^-_{-i}. Formally speaking, each random variable could be larger/smaller from its expectation, which gives in total 2^6 possible ways for those random variables to deviate from their expectations. Nevertheless, there are some trivial cases; for example, it follows from our analysis above that Z_i gets larger if X^+_{+i} gets larger. Hence, we only have to consider the case where X^+_{+i} and X^-_{+i} are smaller than their expectations, and similarly X^+_{-i} and X^-_{-i} are larger than their expectations. Recall also that Y_+ (resp., Y_-) is the number of vertices having wrong belief (resp., correct belief) w.r.t. our expecting planted solution after the first iteration. Thus, for the worst case situation, it would be natural to consider the case where Y_+ is larger and Y_- is smaller than their expectations. Therefore, for estimating the probability that Z_i deviates from its expectation, we introduce small positive parameters σ_1, σ_2, γ_1, ..., γ_4, and consider the following situation as a typical case. (Precisely speaking, we should consider the other three cases for Y_+ and Y_-; but since they can be analyzed similarly, the analyses for these cases are omitted here.)

$$\begin{aligned}
Y_+ \;&= \mathrm{E}[Y_+] + \sigma_1 n' = (r+\sigma_1)n',\\
Y_- \;&= \mathrm{E}[Y_-] - \sigma_2 n' = (p-\sigma_2)n',\\
X^+_{+i} &= \mathrm{E}[X^+_{+i}|Y_+ = (r+\sigma_1)n'] - \gamma_1(r+\sigma_1)n'\\
&= -(p(r+\sigma_1) + \gamma_1(r+\sigma_1))n',\\
X^-_{+i} &= \mathrm{E}[X^-_{+i}|Y_- = (p-\sigma_2)n'] - \gamma_2(p-\sigma_2)n'\\
&= -(r(p-\sigma_2) + \gamma_2(p-\sigma_2))n',\\
X^+_{-i} &= \mathrm{E}[X^+_{-i}|Y_+ = (r+\sigma_1)n'] + \gamma_3(r+\sigma_1)n'\\
&= -(r(r+\sigma_1) - \gamma_3(r+\sigma_1))n',\\
X^-_{-i} &= \mathrm{E}[X^-_{-i}|Y_- = (p-\sigma_2)n'] + \gamma_4(p-\sigma_2)n'\\
&= -(p(p-\sigma_2) - \gamma_4(p-\sigma_2))n'.
\end{aligned} \tag{6}$$

By ignoring positive deviations, we can bound Z_i as follows under this situation.

$$Z_i \;\geq\; (p-r)^2n' - (\sigma_1+\sigma_2)pn' - \gamma_2pn' - \gamma_4pn' - \gamma_1(r+\sigma_1)n' - \gamma_3(r+\sigma_1)n'. \tag{7}$$

Then in order to show that $Z_i > 0$ with high probability, it suffices to show that

$$\max\Big(\sigma_1p,\ \sigma_2p,\ \gamma_2p,\ \gamma_4p,\ \gamma_1(r+\sigma_1),\ \gamma_3(r+\sigma_1)\Big) \;<\; \frac{(p-r)^2}{6} \tag{8}$$

holds with high probability, which is analyzed by the next lemma.

Lemma 3.3. *There is a sufficiently large constant c'_{algo} such that for any $n' > 0$, the probability that the bound (8) does not hold is at most δ' if $p/r \geq c'_{\mathrm{algo}}$ and the following holds*

$$n' \;\geq\; \frac{c'_{\mathrm{algo}}\ln(6/\delta')}{p^2}. \tag{9}$$

Proof. To show the bound (8) holds with the desired probability, we analyze, for each one of the six arguments of $\max(\cdots)$, the probability that it is at least $(p-r)^2/6$.

First consider $\sigma_1 p$ and $\sigma_2 p$. Recall, that $Y_+ \sim \mathrm{B}(n', r)$ and $Y_- \sim \mathrm{B}(n', p)$ and that σ_1 and σ_2 are the deviations of Y_+ and Y_- from their averages. Thus, by Proposition 1.1, if p/r is sufficiently large, we can analyze the probabilities that the desired bounds do not hold, and we can show that those probabilities are indeed smaller than $\delta'/6$ if (9) holds. Precisely, since $Y_+ = rn' + \sigma_1 n'$,

$$\begin{aligned}\Pr\left\{\sigma_1 p \geq \frac{(p-r)^2}{6}\right\} &= \Pr\left\{\frac{Y_+ - rn'}{n'}p \geq \frac{(p-r)^2}{6}\right\}\\ &= \Pr\left\{Y_+ - rn' \geq \frac{(p-r)^2}{6pr}rn'\right\}.\end{aligned}$$

For $(p-r)^2/(6pr) > 1$, by Proposition 1.1(3),

$$\begin{aligned}\Pr\left\{\sigma_1 p \geq \frac{(p-r)^2}{6}\right\} &= \Pr\left\{Y_+ \geq \left(1 + \frac{(p-r)^2}{6pr}\right)rn'\right\}\\ &\leq \exp\left(-\frac{(p-r)^2}{6pr}\frac{rn'}{3}\right)\\ &\leq \exp\left(-\frac{(p-r)^2}{18p^3}c'_{\mathrm{algo}}\ln(6/\delta')\right),\end{aligned}$$

if (9) holds. The value of the last formula above is smaller than $\delta'/6$ if c'_{algo} is sufficiently large. On the other hand, for $(p-r)^2/(6pr) \leq 1$, by Proposition 1.1(2), the probability is at most

$$\begin{aligned}\exp\left(-\frac{(p-r)^4}{36p^2r^2}\frac{rn'}{3}\right) &\leq \exp\left(-\frac{(p-r)^4}{108p^2}n'\right)\\ &\leq \exp\left(-\frac{(p-r)^4}{108p^4}c'_{\mathrm{algo}}\ln(6/\delta')\right),\end{aligned}$$

if (9) holds. The value of the last formula above is smaller than $\delta'/6$ if c'_{algo} is sufficiently large. (The analysis for $\sigma_2 p$ is similar except that Proposition 1.1(1) is used instead.)

Next consider the other four arguments of $\max(\cdots)$. Since the analysis is similar, we state on $\gamma_1(r+\sigma_1)$ as well as $\gamma_2 p$. (The others are similarly derived from these two.) We start with the first one that the probability that $\gamma_1(r+\sigma_1) \geq (p-r)^2/6$ holds is bounded by $\delta'/6$ for n' and p satisfying the condition (9). Noting that we are given $\sigma_1 < p/6$ because of $\sigma_1 \leq (p-r)^2/(6p)$ from the previous analysis, our task is to bound the following probability, for any $\sigma_1 < p/6$, and for each possible value W of W_+ such that $\|W\| = (r+\sigma_1)n'$:

$$\Pr\left[\gamma_1(r+\sigma_1) \geq \frac{(p-r)^2}{6} \;\middle|\; W_+ = W\right].$$

Recall that γ_1 is defined so that $X^+_{+i} = -\mu - \gamma_1(r+\sigma_1)n'$ holds, where

$$\mu \;=\; -\mathrm{E}[X^+_{+i} | Y_+ = (r+\sigma_1)n'] \;=\; -\mathrm{E}[X^+_{+i} | W_+ = W] \;=\; p(r+\sigma_1)n'.$$

Thus, we can restate the above by

$$\begin{aligned}\Pr\left[X^+_{+i} \leq -\mu - \frac{(p-r)^2 n'}{6} \;\middle|\; W_+ = W\right] &= \Pr\left[\sum_{+j\in W_+} -E_{+i,+j} \leq -\left(1 + \frac{(p-r)^2n'}{6\mu}\right)\mu \;\middle|\; W_+ = W\right]\\ &= \Pr\left[\sum_{+j\in W} E_{+i,+j} \geq \left(1 + \frac{(p-r)^2n'}{6\mu}\right)\mu \;\middle|\; W_+ = W\right].\end{aligned}$$

Now a crucial point is the independence of the random variable X^+_{+i} (or, the random variables $E_{+i,+j}, j \in W$) and the event W_+ is this particular set W of size $(r+\sigma_1)n'$ for some $\sigma_1 < p/6$. Recall that W_+ is determined by the first iteration and it depends only on the value of $E_{+j',-1}$ for all $j' \in \{2,\ldots,n\}$, whereas X^+_{+i} depends only on $E_{+i,+j}, j \in \{2,\ldots,n\}$. Thus, by our random model, we may assume that they are independent. (Note that this simple independence argument holds only up to the second iteration; this is why we consider the case that MAXSTEP = 2.) From this observation, we can apply Proposition 1.1(2), (3). For $(p-r)^2n'/(6\mu) > 1$, we have

$$\Pr\left[\sum_{+j\in W} E_{+i,+j} \geq \left(1 + \frac{(p-r)^2n'}{6\mu}\right)\mu \;\middle|\; W_+ = W\right] \leq \exp\left(-\frac{(p-r)^2}{6\mu}\frac{\mu}{3}n'\right)$$

$$\leq \exp\left(-\frac{(p-r)^2}{18p^2}c'_{\mathrm{algo}}\ln(6/\delta')\right),$$

if (9) holds. The value of the last formula above is smaller than $\delta'/6$ if c'_{algo} is sufficiently large. On the other hand, for $(p-r)^2n'/(6\mu) \leq 1$, we have

$$\begin{aligned}
&\Pr\left[\sum_{+j\in W} E_{+i,+j} \geq \left(1+\frac{(p-r)^2n'}{6\mu}\right)\mu \;\middle|\; W_+ = W\right] \\
&\leq \exp\left(-\left(\frac{(p-r)^2n'}{6\mu}\right)^2\cdot\frac{\mu}{3}\right) = \exp\left(-\frac{(p-r)^4n'}{3\cdot 36p(r+\sigma_1)}\right) \\
&\leq \exp\left(-\frac{(p-r)^4n'}{3\cdot 36p^2}\right) \quad (\because\ r+\sigma_1 \leq r+p/6 \leq p) \\
&\leq \exp\left(-\frac{(p-r)^4}{p^4}\cdot\frac{c'_{\mathrm{algo}}\ln(6/\delta')}{108}\right),
\end{aligned}$$

if (9) holds. The value of the last formula above is smaller than $\delta'/6$ if c'_{algo} is sufficiently large.

Similarly, we argue for the second one, i.e., showing that the probability that $\gamma_2 p \geq (p-r)^2/6$ holds is bounded by $\delta'/6$ for n' and p satisfying the condition (9). As before, our task is to bound the following probability, for any $\sigma_2 < p/6$, and for each possible value W of W_- such that $\|W\| = (p-\sigma_2)n'$:

$$\Pr\left[\gamma_2 p \geq \frac{(p-r)^2}{6} \;\middle|\; W_- = W\right].$$

Recall that γ_2 is defined so that $X^-_{+i} = -\mu - \gamma_2(p-\sigma_2)n'$ holds, where

$$\mu = -\mathrm{E}[X^-_{+i}|Y_- = (p-\sigma_2)n'] = -\mathrm{E}[X^-_{+i}|W_- = W] = r(p-\sigma_2)n'.$$

Thus, we can restate the above by

$$\begin{aligned}
\Pr\left[X^-_{+i} \leq -\mu - \frac{p-\sigma_2}{p}\frac{(p-r)^2n'}{6} \;\middle|\; W_- = W\right] &\leq \Pr\left[X^-_{+i} \leq -\mu - \frac{(p-r)^2n'}{36/5} \;\middle|\; W_- = W\right] \quad (\because\ \sigma_2 \leq p/6) \\
&= \Pr\left[\sum_{-j\in W_-} E_{+i,-j} \geq \left(1+\frac{(p-r)^2n'}{(36/5)\mu}\right)\mu \;\middle|\; W_- = W\right].
\end{aligned}$$

From the observation about independence same as before, we can apply Proposition 1.1(2), (3). For $(p-r)^2n'/((36/5)\mu) > 1$, we have

$$\begin{aligned}
\Pr\left[\sum_{-j\in W_-} E_{+i,-j} \geq \left(1+\frac{(p-r)^2n'}{(36/5)\mu}\right)\mu \;\middle|\; W_- = W\right] &\leq \exp\left(-\frac{(p-r)^2}{(36/5)\mu}\frac{\mu}{3}n'\right) \\
&\leq \exp\left(-\frac{(p-r)^2}{(108/5)p^2}c'_{\mathrm{algo}}\ln(6/\delta')\right),
\end{aligned}$$

if (9) holds. The value of the last formula above is smaller than $\delta'/6$ if c'_{algo} is sufficiently large. On the other hand, for $(p-r)^2n'/((36/5)\mu) \leq 1$,

$$\begin{aligned}
\Pr\left[\sum_{-j\in W_-} E_{+i,-j} \geq \left(1+\frac{(p-r)^2n'}{(36/5)\mu}\right)\mu \;\middle|\; W_- = W\right] &\leq \exp\left(-\frac{(p-r)^4}{(36/5)^2\mu^2}\frac{\mu}{3}n'^2\right) \\
&= \exp\left(-\frac{(p-r)^4}{3(36/5)^2\mu}n'^2\right) \\
&\leq \exp\left(-\frac{(p-r)^4}{3(36/5)^2p^2}n'\right) \quad (\because\ \mu \leq p^2n') \\
&\leq \exp\left(-\frac{(p-r)^4}{3(36/5)^2p^4}c'_{\mathrm{algo}}\ln(6/\delta')\right) \leq \delta'/6.
\end{aligned}$$

With similar analysis for the other two bounds, we conclude that the probability that some of the bounds does not hold is bounded by $6*(\delta'/6) = \delta'$. □

In summary, if the bound (8) holds, then we have $Z_i > 0$, which means that the algorithm outputs $+1$ for x_i; thus, if the same situation holds for every $Z_{i'}$, $2 \leq i' \leq n$, then the algorithm yields the all $+1$ planted solution. Therefore, using the above lemma with $\delta' = \delta/2n < \delta/2n'$ and by bounding the total error by the union bound, we prove Theorem 3.1.

Acknowledgements

We are grateful to an anonymous referee for his/her careful and persistent checking of our manuscript.

The first author supported in part by a Grant-in-Aid for Scientific Research on Priority Areas "New Horizons in Computing" 2004–2006.

References

[1] D. Achlioptas, H. Jia, C. Moore, Hiding satisfying assignments: two are better than one, Journal of the AI Research 24 (2005) 623–639. (Preliminary version, in: Proc. 19th Natl. Conf. on Artificial Intelligence AAAI '04, pp. 131–136).
[2] D. Boughaci, H. Drias, Efficient and experimental meta-heuristics for MAX-SAT problems, in: Proc. Workshop on Experimental/Efficient Algorithms, in: Lecture Notes in Computer Science, vol. 3503, 2005, pp. 501–512.
[3] H. Drias, Scatter search with walk strategy for solving hard Max-W-Sat problems, in: Proc. of Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, in: Lecture Notes in Compute Science, vol. 2070, 2001, pp. 35–44.
[4] J. Håstad, Some optimal inapproximability results, Journal of the ACM 48 (2001) 798–859.
[5] B. Mazure, L. Sais, E. Greroire, A tabu search for SAT, in: Proc. of the 15th National Conf. on Artificial Intelligence, AAAI97, 1997, pp. 281–285.
[6] C. McDiarmid, Concentration, in: M. Habib, C. McDiarmid, J. Ramirez-Alfonsin, B. Reed (Eds.), Probabilistic Methods for Algorithmic Discrete Mathematics, Springer, 1998.
[7] R. McEliece, D. MacKay, J. Cheng, Turbo decoding as an instance of Pearl's Belief Propagation algorithm, IEEE Journal on Selected Areas in Communications 16 (2) (1998).
[8] M. Motoki, Random instance generation for MAX 3SAT, in: Proc. of 7th Annual Int'l Computing and Combinatorics Conference, COCOON'01, in: Lecture Notes in Computer Science, vol. 2108, 2001, pp. 502–508.
[9] M. Motoki, Test instance generation for MAX 2SAT, in: Proc. of 11th Int'l Conf. on Principles and Practice of Constraint Programming, CP'05, in: Lecture Notes in Computer Science, vol. 3709, 2005, pp. 787–791.
[10] R. Motwani, P. Raghavan, Randomized Algorithms, Cambridge University Press, 1995.
[11] M. Onsjö, O. Watanabe, Simple algorithms for graph partition problems, Research Report C-212, Dept. of Math. and Comput. Sci., Tokyo Inst. of Tech., 2005.
[12] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann Publishers Inc, 1988.
[13] A.D. Scott, G.B. Sorkin, Faster algorithms for MAX CUT and MAX CSP, with polynomial expected time for sparse instances, in: Proc. APPROX and RANDOM 2003, in: LNCS, vol. 2764, 2003, pp. 382–395.
[14] B. Selman, H.J. Levesque, D.G. Mitchell, A new method for solving hard satisfiability problems, in: Proc. of the 10th National Conf. on Artificial Intelligence, AAAI92, 1992, pp. 440–446.
[15] M. Yamamoto, Generating instances for MAX2SAT with optimal solutions, Theory of Computing Systems 39 (5) (2006) 723–742.